# Online Transaction Fraud Detection Project Report

The project classifies fraudulence of online transactions. Initially we have downloaded the dataset provided. The dataset has been split into training and testing models. It has been joined on the column "TransactionID".

**Feature Selection:**

The following are the steps we implemented while feature selection:

**Reading the CSV :**

- The csv files are read using two data frames: 'train transaction' for the transaction csv file, and 'train identity' for the identity csv file.
- The parameters are selected to show the most rows and columns possible.
- Another data frame named 'train df' is defined to integrate the aforementioned data frames on the 'TransactionID' column, which is shared by both files.
- The train df is further divided into 'train df' and 'test df,' each with a 20% test size.
- For a straightforward analysis, the data frame has ambiguous ('_') and ('-') that are changed to only ('-').

**Memory Reduction:**

- It's a must-do before diving into a massive data set for analysis.
- Because the dataset is too large to run in a Jupyter notebook, https://www.kaggle.com/gemartin/load-data-reduce-memory-usage is used to run the dataset and perform data analysis on it.On each dataframe, it saves between 60 and 75 percent of memory use.
- For memory reduction, a function named 'reduce memory usage()' is declared, and 'train' and 'test' data frames are stored in it to reduce their memory usage.
- To identify null columns, missing values table()' function is declared.
- To improve the precision of the analysis, null columns are sorted in decreasing order and saved as ratios.
- NAN's are used to replace null columns.

**Feature Engineering:**

The following are the steps implemented in feature engineering in our project:

- The date format is the first feature to choose. The time and date are set to a 24-hour format.
- The alerts for hour features are set to High, Medium, or Low depending on the hours provided.
- A list of worthless characteristics is created.

**Exploratory Data Analysis:**

- Dropped useless features
- Converted Infinity values to NAN.
- Dropped the missing variables that have ratios more than 60%.
- Filled Missing values to -999 in train and test dataframe
- Converted the string to integer for the accuracy of algorithms using LabelEncoder().
- The Model was taking time to execute therefore we reduced the memory again

**Modeling:**

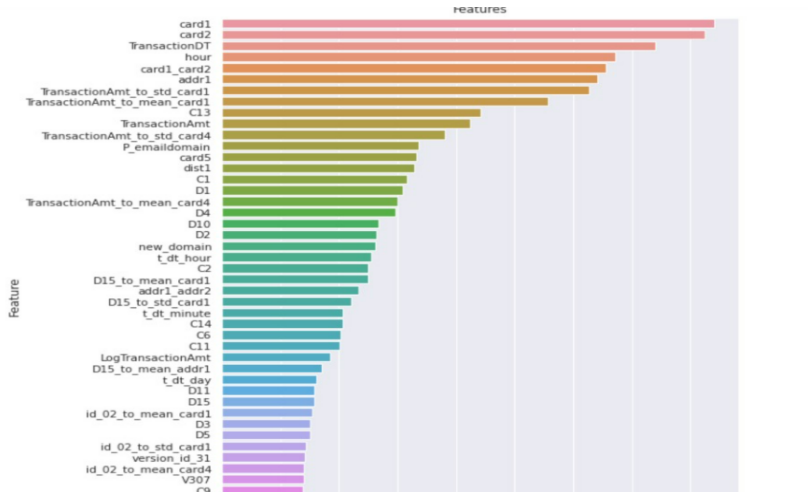
The model was built by dropping the columns of fraud and transaction ID

The model of train and test data was built again .

**Random Under Sampling:**
It was performed in order to balance the imbalance of the training and testing models.

**Plotting:**
We have displayed classification using two machine learning models LGBM and XGB classifiers

**LGBM Classifier :**
Light GBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:
Light GBM is a relatively new algorithm and has a long list of parameters. The size of the dataset is increasing rapidly. It has become very difficult for traditional data science algorithms to give accurate results. Light GBM is prefixed as Light because of its high speed. Light GBM can handle the large size of data and takes lower memory to run. Another reason why Light GBM is so popular is because it focuses on accuracy of results. LGBM also supports GPU learning and thus data scientists are widely using LGBM for data science application development. It is not advisable to use LGBM on small datasets. Light GBM is sensitive to overfitting and can easily overfit small data.

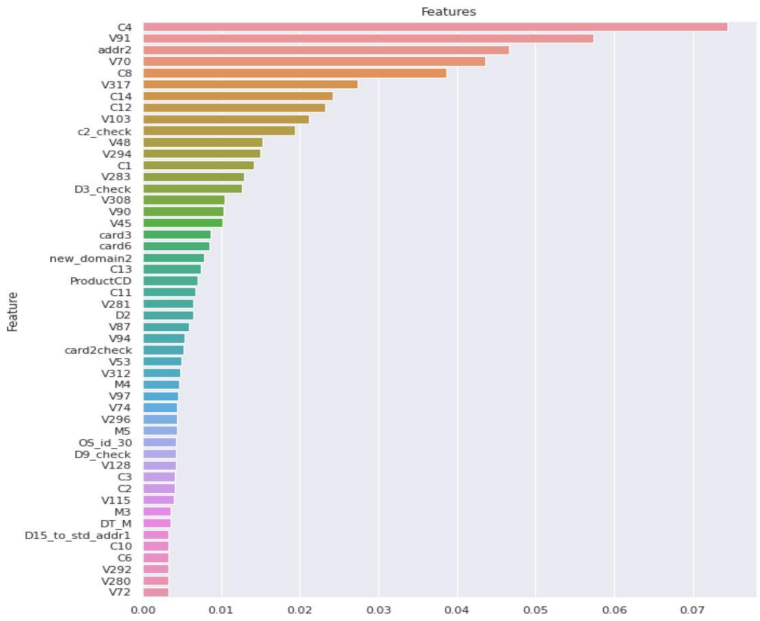



The LGBM classifier displays classification in descending order. The length of the line displays the dependency of a transaction as true or false at the respective feature.

The accuracy of the classifier is 88%.

**XGB Classifier:**

XGBoost (eXtreme Gradient Boosting) is a popular supervised-learning algorithm used for regression and classification on large datasets. It uses sequentially-built shallow decision trees to provide accurate results and a highly-scalable training method that avoids overfitting.




XGB classifier similarly displays a set of features and the colored lines display the fraudulence of a transaction according to their respective features.

XGB classifier displays 87% accuracy for this project.

**LGBM Classifier vs XGB CLassifier :**

- XGBoost Classifier is a very fast and accurate ML algorithm. But now it's been challenged by LightGBM Classifier — which runs even faster with comparable model accuracy and more hyperparameters for users to tune.
- The key difference in speed is because XGBoost split the tree nodes one level at a time and LightGBM does that one node at a time.
- So XGBoost developers later improved their algorithms to catch up with LightGBM, allowing users to also run XGBoost in split-by-leaf mode (grow_policy = ‘lossguide’). Now XGBoost is much faster with this improvement, but LightGBM is still about 1.3X — 1.5X the speed of XGB.
- Another difference between XGBoost and LightGBM is that XGBoost has a feature that LightGBM lacks — monotonic constraint. It will sacrifice some model accuracy and increase training time, but may improve model interpretability.
- Plot importance is changing using these two types of Classification.